LA-UR-15-28099
Approved for public release; distribution is unlimited.

Title:	Beyond Defensive IO: Leveraging the Burst Buffer for In-transit Visualization Workflows
Author(s):	Canada, Curtis Vincent Patchett, John M. Braithwaite, Ryan Karl Ahrens, James Paul Ruiz Varela, Maria
Intended for:	Web
Issued:	2015-10-19

Maria Ruiz Varela
University of Delaware

Data Science at Scale Summer School

Los Alamos National Laboratory

August 28, 2015

Beyond Defensive IO: Leveraging the Burst Buffer for In-transit Visualization Workflows

Maria Ruiz Varela

John Patchett

Ryan Braithwaite

Jim Ahrens

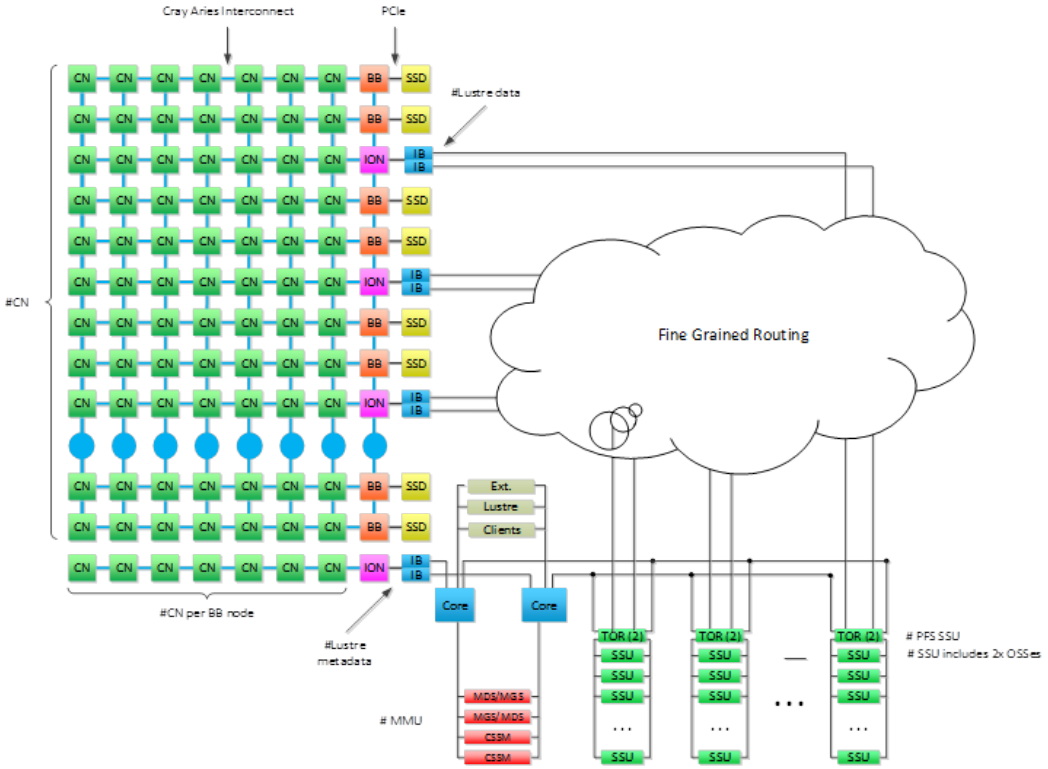
Outline

- Background
- Motivation
- Approach
- Conclusions
- Future work
- Questions

Background

Trinity System

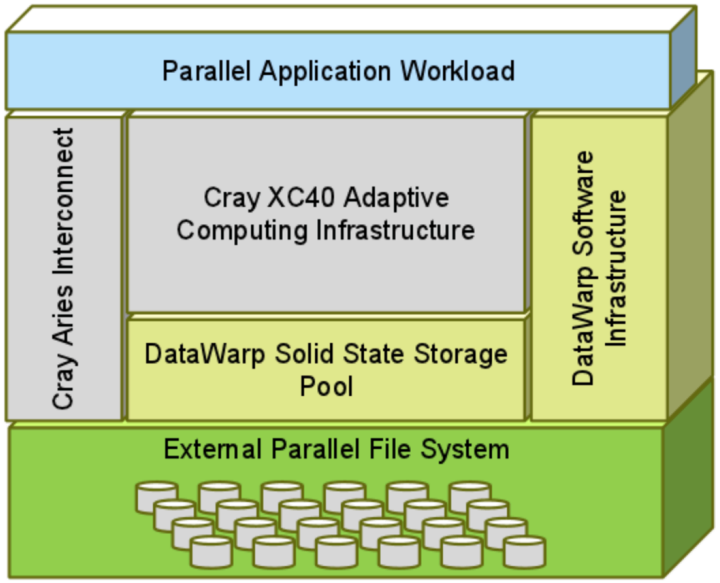
- First advanced technology system of the ASC program
- Will include burst buffers
  - Connected to the high speed network closer to the compute nodes, farther from the the PFS

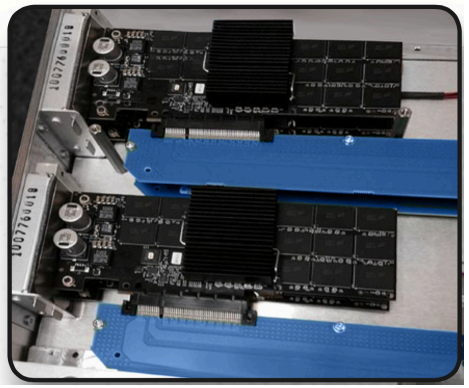


Cray XC40 Data Warp Blade (2 nodes)

http://www.cray.com/sites/default/files/resources/CrayXC40-DataWarp.pdf

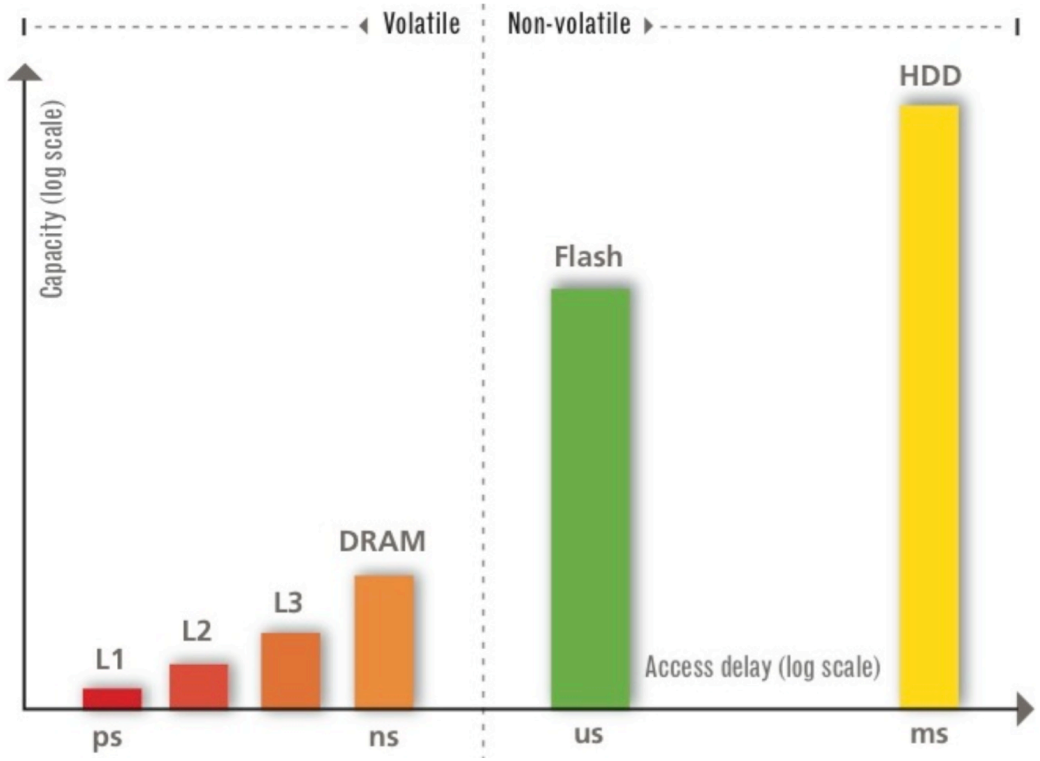
- Technology Drivers:
  - Solid State Disk (SSD) cost decreasing
  - Lower cost of bandwidth than hard disk drive
- Trinity Operational Plans:
  - 3 PB Burst Buffer, SSD based
  - 1.45 TB/Sec (2x speed of Parallel File System)
- Burst Buffers to improve operational efficiency by reducing defensive IO time
- Burst Buffer fills a gap in the Memory and Storage Hierarchy





Gap in the memory hierarchy

From: Fusion-IO Taming the Power Hungry Data Center



Storage	Food	Relative Access Time
L1 cache	Food in the mouth	Fractions of a second
L2 cache	Get food from the plate	1 second
L3 cache	Get food from the table	Few seconds
DRAM	Get food from the kitchen	Few minutes
FLASH	Get food from the neighborhood store	Few hours
HDD	Get food from Mars!	3-5 years

http://www.fusionio.com/white-papers/taming-the-power-hungry-data-center

Motivation

- Given this infrastructure
  - How could applications other than C/R leverage the BB?
  - How could in-transit visualization workflows leverage the BB?
  - Which components of the BB would a visualization workflow use?
    - Schedule compute nodes or BB compute resources?
  - How to ensure applications are prepared for, take advantage of, and execute efficiently with this new layer?
  - We need to collect metrics to:
    - Understand performance, endurance limitations, etc.

In-transit visualization

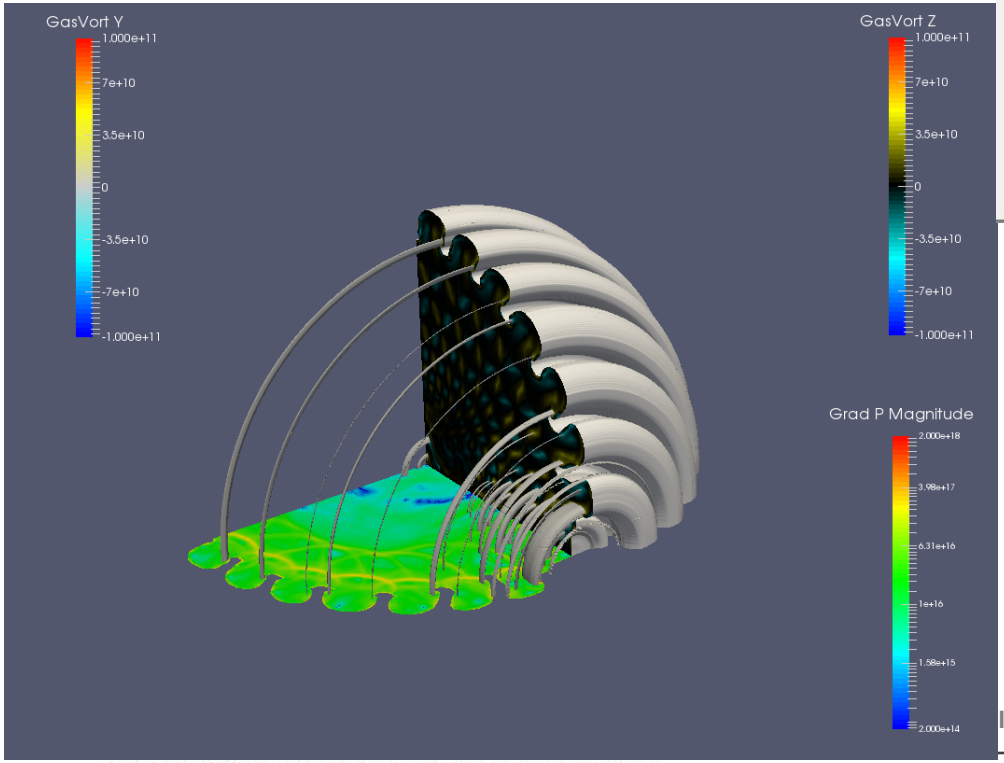
- Simulation produces extremely large data sets
- Prohibitively expensive to save all data to persistent storage and then read it back for analysis
- Alternatives:
  - **In-situ:** Uses the primary compute resource for analysis
  - **In-transit:** Offloads simulation results to secondary resources for processing, while in transit to persistent storage.

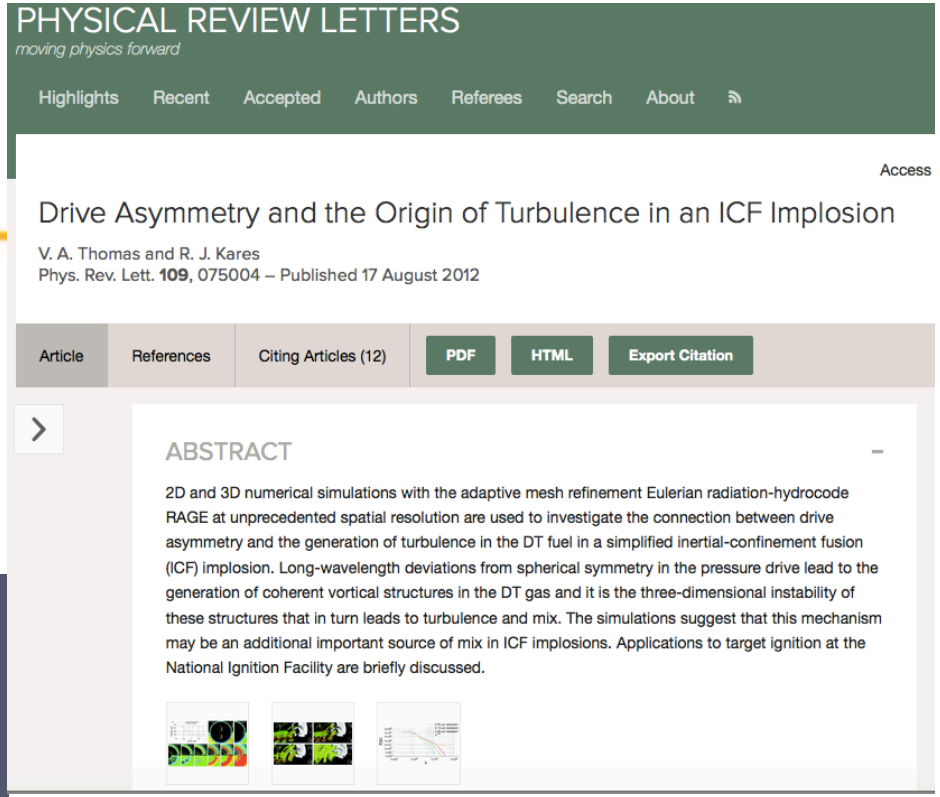
Approach

- We collect metrics to understand performance
  - Assume raw data produced by the simulation is in the burst buffer file system
  - Process the input data in the burst buffer
  - Write the resulting image

Benchmark file

- Assume the raw data file produced by the simulation is in the BB file system
- Input file is 28 GB in size
- Comprised of 256 vtu files
- Read file with VTK (Visualization Toolkit)






```
from paraview.simple import *
from paraview import benchmark
timer = paraview.vtk.vtkTimerLog()

benchmark.maximize_logs()

reader = OpenDataFile("/nifdata/grid_0.vtm")
reader.UpdatePipeline()

benchmark.get_logs()
benchmark.print_logs()
```

Experiment platform – Darwin cluster

- Initial partition

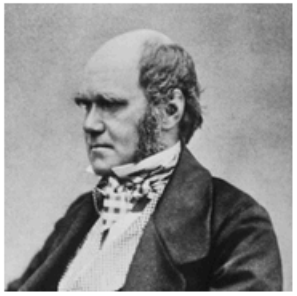
8x HP DL360 gen 8 nodes

- 6 cores
- 16 GB DDR3
- 4x200 GB SSD (SATA)
- 4x500 GB HDD (SATA)
- 1x400 GB Intel P3700 (PCIe)
- 10 Gb Ethernet

- Used partition

8 nodes, 256 cores

- 32 cores
- 128 GB RAM
- 1x200 GB SSD (SATA)
- 1x500 GB HDD (SATA)
- 1x400 GB Intel P3700 (PCIe)
- 10 Gb Ethernet
- NFS (Network File System)



Darwin

Linux Co-design Cluster

Darwin is an ASC prototyping cluster. It contains many different server architectures and accelerators. It is designed to facilitate research and development work for ASC and ASCR projects by coalescing cutting-edge hardware and software into a single cluster to leverage common network and storage resources in a unified OS environment. Because Darwin is a very heterogeneous cluster, many SLURM partitions exist to help users find the type of hardware they're looking for. These partitions are not mutually exclusive, so be prepared to take a little time to become familiar with how to request the type of node you're looking for. One current special partition is a 16 node portion configured as a Burst Buffer test area. Another is a 21 node portion configured with a Hadoop Distributed File System (HDFS). Both of these partitions have been created for the Data Science at Scale team. Darwin is run by CCS-7.

Experiments

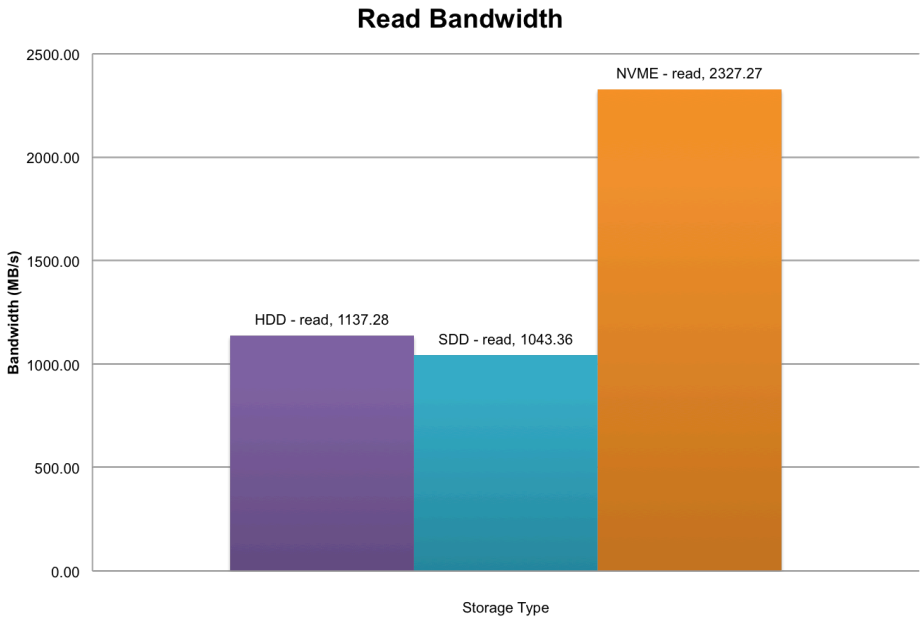
- For each of the three file systems:
  - Measure raw device, sequential read performance
    - dd linux utility, fio benchmark
  - Measure read latency of a read visualization pipeline

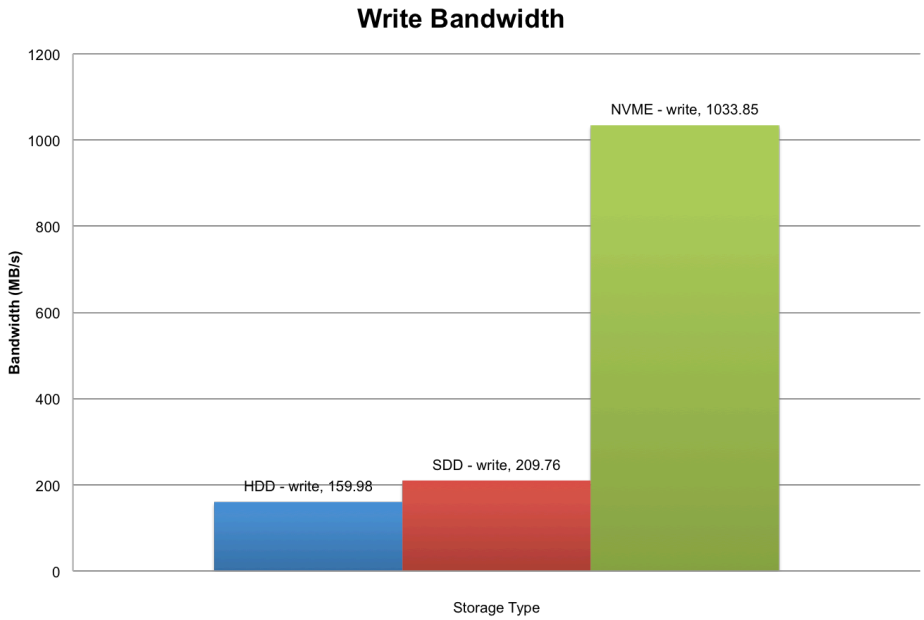
```
from paraview.simple import *
from paraview import benchmark
timer = paraview.vtk.vtkTimerLog()

benchmark.maximize_logs()

reader = OpenDataFile("/nifdata/grid_0.vtm")
reader.UpdatePipeline()

benchmark.get_logs()
benchmark.print_logs()
```

  - Measure analysis time for varying number of processes
    - mpirun -np 8 --npernode 1 pvbatch nifscript.py
    - mpirun -np 16 --npernode 2 pvbatch nifscript.py

Results

- Sequential read and write performance



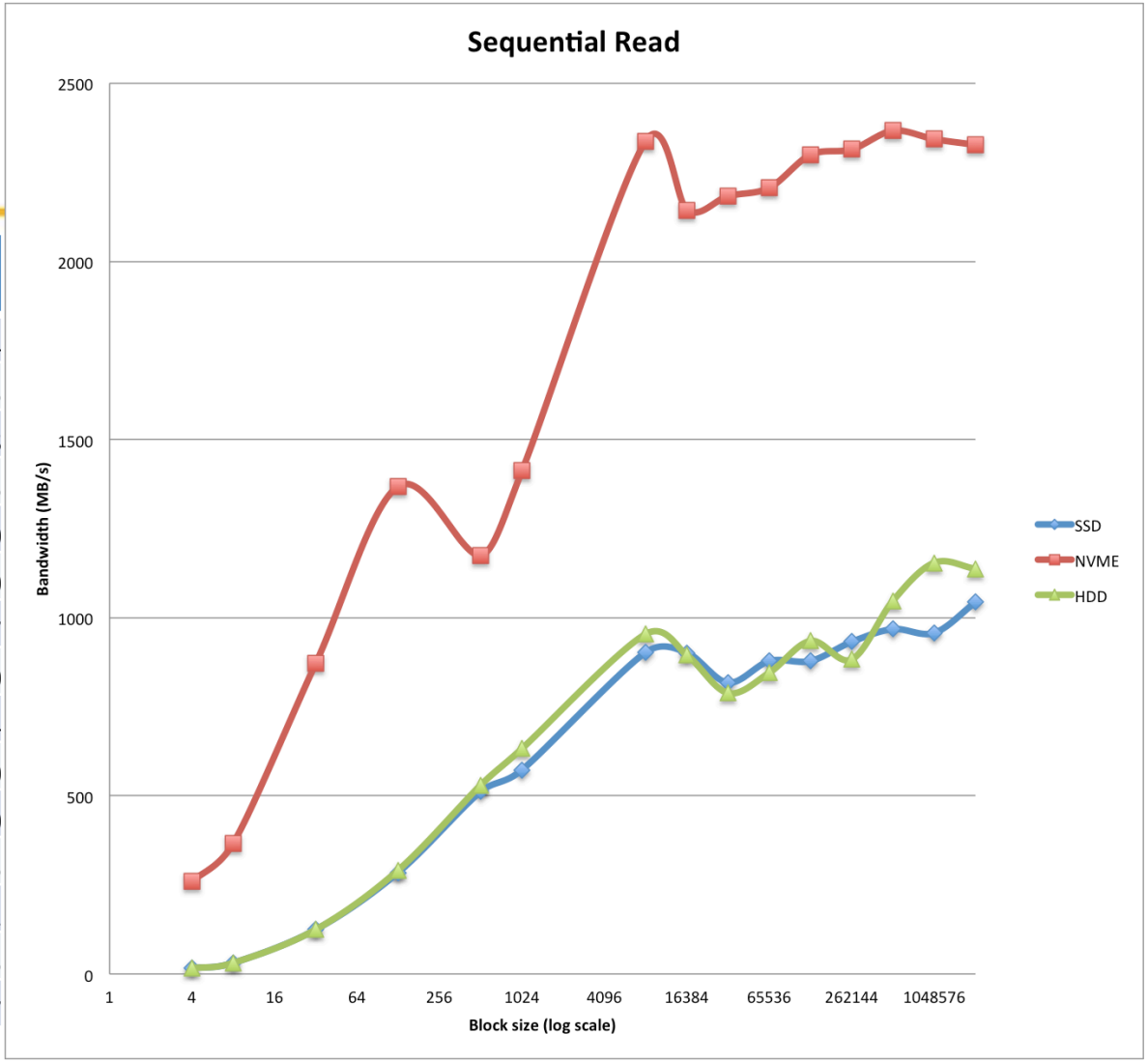


Fio, 2 GB file, direct = 1, bs = 2 GB

Fio, 2 GB file, direct = 1, bs = 8 MB

Results

Data point	Block Size (KB)	HDD BW (MB/s)	SSD BW (MB/s)	NVME BW (MB/s)
1	4	15.59	15.59	260.04
2	8	31.19	31.19	367.15
3	32	124.66	124.49	871.88
4	128	291.36	284.79	1368.75
5	512	530.14	512.00	1175.00
6	1024	632.45	572.94	1412.00
7	8192	954.07	902.65	2337.97
8	16384	894.48	901.35	2144.00
9	32768	788.42	817.55	2183.24
10	65536	845.70	878.06	2206.90
11	131072	934.74	878.62	2299.40
12	262144	883.00	932.36	2314.65
13	524288	1047.70	967.78	2366.72
14	1048576	1154.28	957.34	2343.25
15	2097152	1137.28	1043.36	2327.27



- Fio benchmark, 2GB file, direct = 1, bs = [4 KB .. 2 GB]

Results

NVME

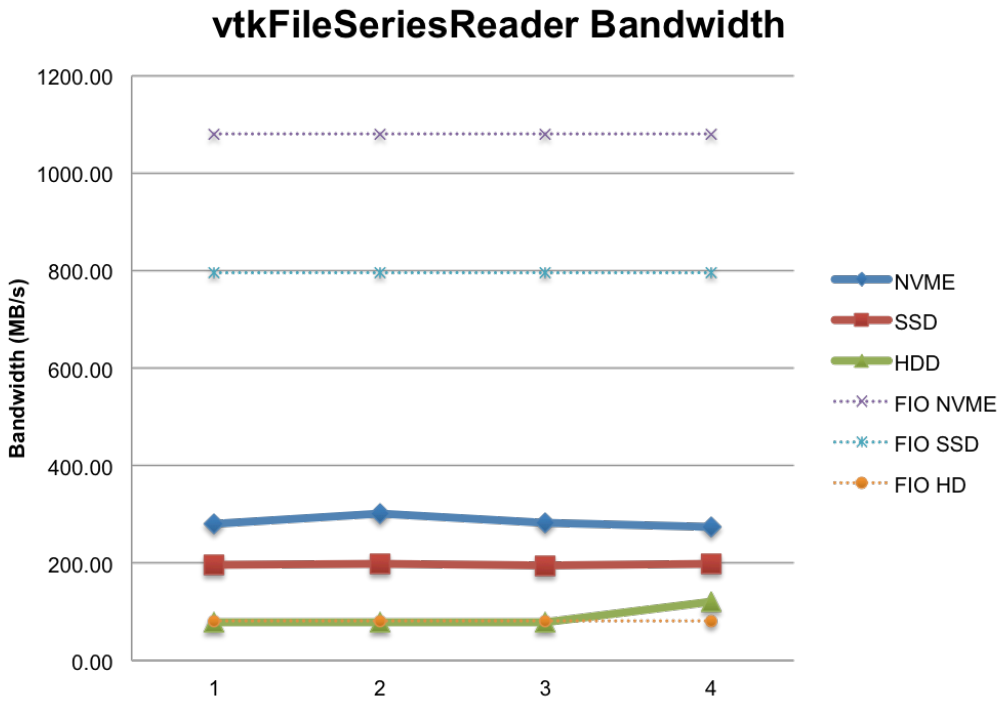
	Time ParaView Reader	Bandwidth ParaView Reader
Run 1	102.00 sec	281.10 MB/s
Run 2	95.00 sec	301.81 MB/s
Run 3	101.62 sec	282.15 MB/s
Run 4	104.74 sec	273.74 MB/s
Avg	100.84	

SSD

	Time ParaView Reader	Bandwidth ParaView Reader
Run 1	196.94 sec	145.59 MB/s
Run 2	199.10 sec	144.01 MB/s
Run 3	195.23 sec	146.86 MB/s
Run 4	198.18 sec	144.68 MB/s
Avg	197.36	

HDD

	Time ParaView Reader	Bandwidth ParaView Reader
Run 1	366.48 sec	78.24 MB/s
Run 2	366.37 sec	78.26 MB/s
Run 3	366.19 sec	78.30 MB/s
Run 4	237.04 sec	120.96 MB/s
Avg	334.02	



Other (partial) results

- Analysis time
  - mpirun -np 8 --npernode 1 pvbatch nifscript.py
  - mpirun -np 16 --npernode 2 pvbatch nifscript.py

	n = 8	n = 16
Run 1	1,487 s ~ 25 min	786 s ~ 13 min
Run 2	1,448 s ~ 24 min	748 s ~ 12 min

Conclusions

- Applications and systems need to be modified to be:
  - Aware of new layers in the memory hierarchy
  - Underlying device and file system combination
- Performance factors
  - Block size: *Example of fixed, hard-coded, device-oblivious block size in code*

```
~/Software/VTK-6.2.0/IO/XMLParser/vtkXMLParser.cxx
//Default stream parser just reads a block at a time.
istream& in = *(this->Stream);
const int bufferSize = 4096;
char buffer[bufferSize];
```

  - Memory alignment
  - Number of posix calls made

Future work

- Time all components of the read, analyze, write pipeline
- Complete measurements of read latency vs. number of processes
- Measure IO power consumption for the 3 devices
  - Watt meter (Watts up) already connected to node cn119
- Characterize applications to understand implications of limited lifetime of flash devices for different types of workloads
  - Read vs. write intensive
- Burst buffers can reduce IO wait time
  - Bent, John, et al. "Jitter-free co-processing on a prototype exascale storage stack." Mass Storage Systems and Technologies (MSST), 2012 IEEE 28th Symposium on. IEEE, 2012.
  - But could they increase power consumption?

Acknowledgements

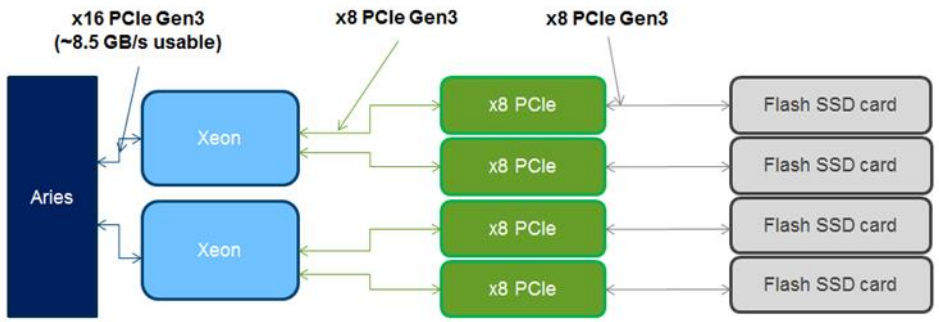
- Jim, John Patchett, Curt, Erika, Robyn, DSSS mentors and students, Boonth, Ryan, Chris Mitchell, Noah Watkins

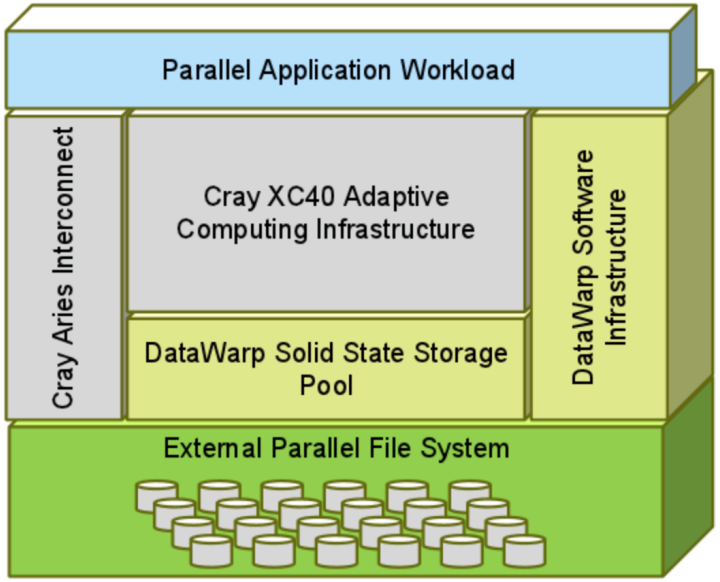
Questions?

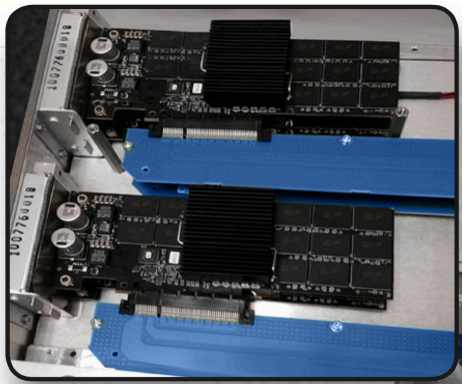
Additional slides

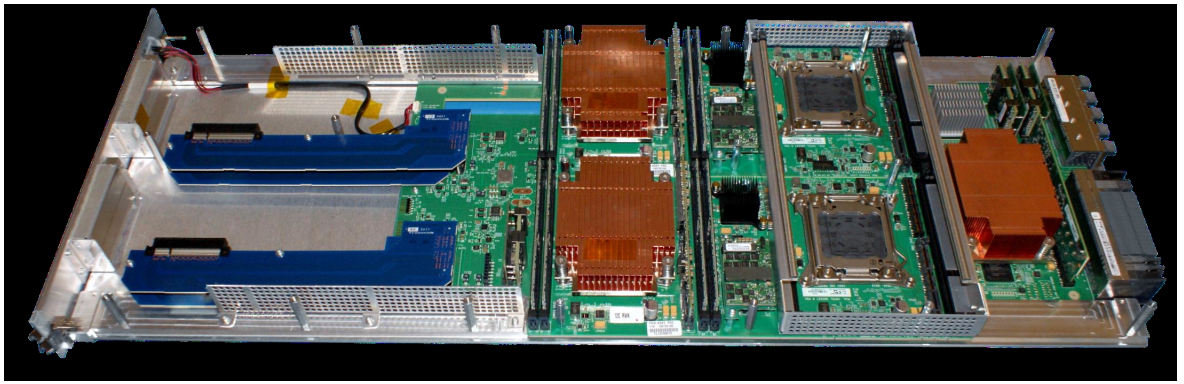
Cray XC40 DataWarp Blade (2 nodes)

http://www.cray.com/sites/default/files/resources/CrayXC40-DataWarp.pdf









Trinity Burst Buffer Hardware

- Trinity
  - ~10K Haswell + ~10K KNL nodes
  - 2.1 PB memory
- 576 Burst Buffer Nodes
  - Announced as Cray DataWarp™
  - On high speed interconnect
  - Globally accessible
  - Trinity IO Node + PCIe SSD Cards
  - Distributed throughout cabinets

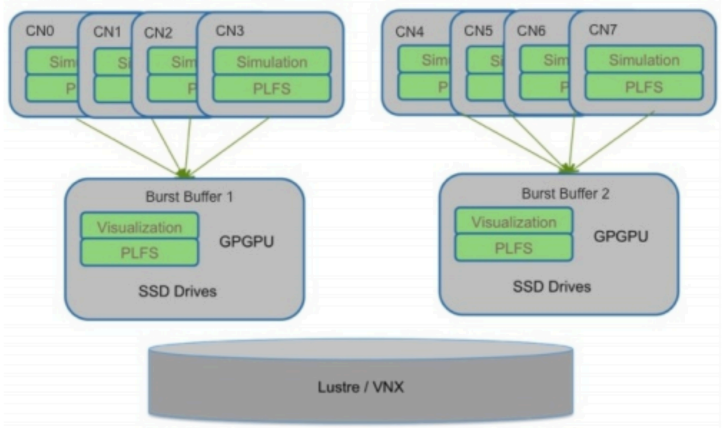
Metric	Burst Buffer	PFS
Nodes	576 BB Nodes	234 LNET Routers
Bandwidth	3.3 TB/S	1.45 TB/S
Capacity	3.7 PB	82 PB
Memory Multiple	1.75 X	39 X
Application Efficiency	88%	79%
App Time Writing CR	12%	21%

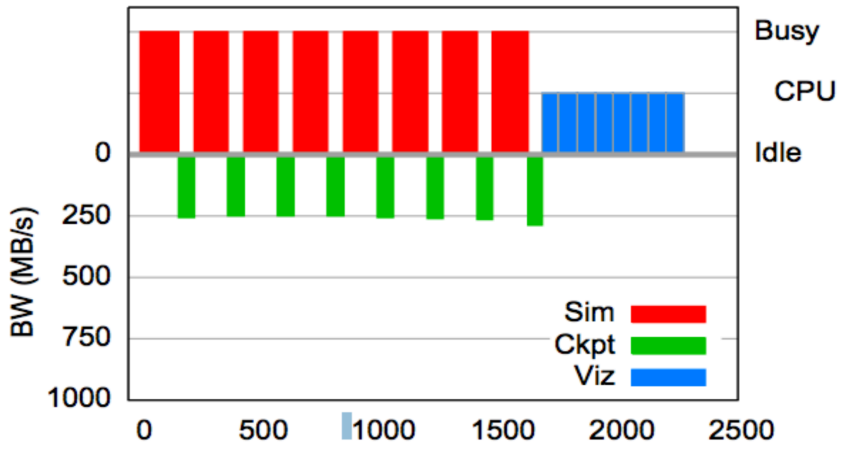
Bridging the Gap – LDRD Proposal in Submission

- How do we leverage the Burst Buffer to get Science Done?
  - Extended available memory space
    - Treating burst buffer as memory
    - Explore out-of-core algorithms
  - Co-scheduling of data motion and compute
    - Automate data movement in deep memory hierarchies
  - Data representations and algorithms
    - Explore new data representations and their mapping to the memory hierarchy
    - Optimize for performance and scalability

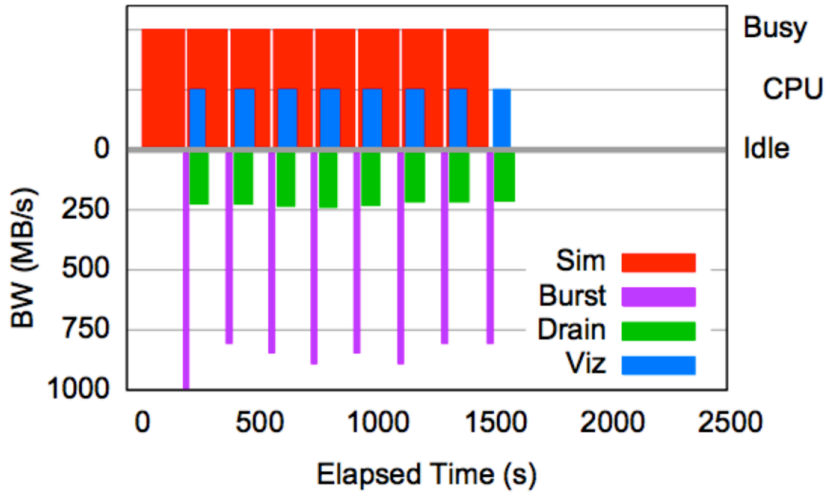
Jitter Free Coprocessing on a Prototype exascale storage stack

Bent, John, et al. "Jitter-free co-processing on a prototype exascale storage stack." Mass Storage Systems and Technologies (MSST), 2012 IEEE 28th Symposium on. IEEE, 2012.





(a) Direct to Lustre



(b) Using Burst Buffers

Data point	Block size	HDD	SSD	NVME
1	4	15.59	15.59	260.04
2	8	31.19	31.19	367.15
3	32	124.66	124.49	871.88
4	128	291.36	284.79	1368.75
5	512	530.14	512.00	1175.00
6	1024	632.45	572.94	1412.00
7	8192	954.07	902.65	2337.97
8	16384	894.48	901.35	2144.00
9	32768	788.42	817.55	2183.24
10	65536	845.70	878.06	2206.90
11	128	934.74	878.62	2299.40
12	256	883.00	932.36	2314.65
13	512	1047.70	967.78	2366.72
14	1024	1154.28	957.34	2343.25
15	2048	1137.28	1043.36	2327.27

Outline

- Conclusions
- Future work
  - Second level text
    - Third level text
      - Fourth level text

Slide Title

- Bulleted text
- More bulleted text
- And more bulleted text

Insert chart, picture, etc., here

The caption goes here

Slide Title

- Bulleted text
- More bulleted text
- And more bulleted text

Insert chart, picture, etc., here

The caption goes here

Slide Title

- Bulleted text
- More bulleted text
- And more bulleted text

Insert chart, picture, etc., here

The caption goes here